LAB 8

STAT 28

March 15, 2017

This is a long lab with 16 pages (what !?). But do not get intimidated because most of the pages are just plots. In this lab, you will learn some visualization methods in R which include:

- Heatmap
- PCA
- A brief introduction to `ggplot`

The last section is optional. If you are already very comfortable with basic R plotting, you are welcomed to explore the visualization tool `ggplot`.

Place rated dataset

The data is taken from a book named **Places Rated Almanac**. They used nine rating criteria to rate cities. For all but two of the criteria, the higher the score, the better. For Housing and Crime, the lower the score the better. The scores are computed using the following component statistics for each criterion:

- Climate & Terrain: very hot and very cold months, seasonal temperature variation, heating- and cooling-degree days, freezing days, zero-degree days, ninety-degree days.
- Housing: utility bills, property taxes, mortgage payments.
- Health Care & Environment: per capita physicians, teaching hospitals, medical schools, cardiac rehabilitation centers, comprehensive cancer treatment centers, hospices, insurance/hospitalization costs index, flouridation of drinking water, air pollution.
- Crime: violent crime rate, property crime rate.
- Transportation: daily commute, public transportation, Interstate highways, air service, passenger rail service.
- Education: pupil/teacher ratio in the public K-12 system, effort index in K-12, accademic options in higher education.
- The Arts: museums, fine arts and public radio stations, public television stations, universities offering a degree or degrees in the arts, symphony orchestras, theatres, opera companies, dance companies, public libraries.
- Recreation: good restaurants, public golf courses, certified lanes for tenpin bowling, movie theatres, zoos, aquariums, family theme parks, sanctioned automobile race tracks, pari-mutuel betting attractions, major- and minor- league professional sports teams, NCAA Division I football and basketball teams, miles of ocean or Great Lakes coastline, inland water, national forests, national parks, or national wildlife refuges, Consolidated Metropolitan Statistical Area access.
- Economics: average household income adjusted for taxes and living costs, income growth, job growth.

Read Data.

```
place_rated <- read.csv("place.csv", stringsAsFactors = FALSE)
place_rated[, 1:9] <- scale(place_rated[, 1:9])
CA_NY_cities <- which(place_rated$state %in% c("CA", "NY"))
example_cities <- CA_NY_cities[c(10, 13, 15, 22, 24, 25, 26)]
row.names(place_rated) <- paste0(place_rated$city, place_rated$state)
```

Heatmap

Exercise 1

We learned two types of heatmaps in lecture 4. One is heatmap for correlation matrix, the other is heatmap for data matrix.

(a) Calculate the correlation matrix between nine rating criteria using function `cor`. What is the correlation between arts and education?

```
# insert your code here and save the correlation matrix as
# `cor_place_rated`
# cor_place_rated <-
# read the output from `cor_place_rated`, copy and paste your answer below
# cor_arts_edu <-
```

(b) Plot the heatmap for the correlation matrix using the `aheatmap` from `NMF` package. If you are going to divide the nine rating criteria into two categories based on similarity, how would you split the variables?

```
# insert your code here for heatmap
library(NMF)
```

```
## Loading required package: pkgmaker
```

```
## Loading required package: registry
```

```
##
## Attaching package: 'pkgmaker'
```

```
## The following object is masked from 'package:base':
##
##     isNamespaceLoaded
```

```
## Loading required package: rngtools
```

```
## Loading required package: cluster
```

```
## NMF - BioConductor layer [OK] | Shared memory capabilities [NO: bigmemory] | Cores 3/4
```

```
##   To enable shared memory capabilities, try: install.extras('
## NMF
## ')
```

(c) Plot the heatmap for data matrix using the `aheatmap` from `NMF` package.

```
# insert your code here for the heatmaps
```

PCA

Exercise 2

The principal component analysis allows you to convert a set of correlated variables into a set of linearly uncorrelated principal components. Each principal component is a linear combination of the original variables. The first principal component has the largest variance and the last principal component the least. You can think that the first principal component carries most information of the data (explains the largest proportion of data variance). That is why we usually use the first and the second PC to do visualization.

(a) Run PCA on nine rating criteria using function `place.pca`.

```
# insert your code here to run pca, save your result as `place.pca`
# place.pca <-
```

(b) Create a scatter plot of the first principal component versus the second principal component.

HINT: Similar to `lm`, `prcomp` gives you a list of objects. They are `sdev`, `rotation`, `center`, `scale` and `x`. You can access them using the dollar sign `$`. For now, you do not need to worry about what they are except `x`, which gives you a matrix with 9 columns (PCs). You may create scatter plot of any two PCs using this matrix.

```
# insert your code here for the scatter plot
```

For this data set, we do not have any categories associated with each sample. Thus, we are not able to observe the separation between groups when plotting the first two PCs (as you've seen in class). However, we can still observe something interesting by using the `biplot` function as follows. Besides creating scatter plot between PCs, `biplot` function also plots the weights of the linear combinations (loadings) when calculating principal components. We marked several cities in the `biplot` output as follows.

```
# # Uncomment the code below after you get `place.pca`
# text_names = place_rated$city
# text_names[-example_cities] = 'o'
# biplot(place.pca, cex = 0.5, xlabs = text_names, xlim = c(-0.15, 0.4), ylim = c(-0.2, 0.2))
```

(c) Use `summary` function on your `place.pca` object. How many PCs will you choose in order to explain at least 80% of the data variance?

```
# insert your code here to summary your PCs
#
```

```
# How many PCs will you choose in order to explain at least 80% of the data variance?
# Please uncomment your answer
# nPC <- 2
# nPC <- 4
# nPC <- 5
# nPC <- 8
```

A Brief Introduction to ggplot (Optional)

`ggplot` is probably the most widely used plotting package in R. It creates nice looking graphics with much cleaner and easier arguments (compare to functions in the base package). It is very user-friendly especially when dealing with datasets with categorical variables. `ggplot` allows you to create plots layer by layer, making it possible to create sophisticated plots.

In this section, we will create several plots using `ggplot` with our familiar craigslist dataset. These are very simple examples of what ggplot can do. If you are interested, you can refer to the document of `ggplot`(http://docs.ggplot2.org/current/).

```
library(ggplot2)
craigslist <- read.csv("craigslist.csv")
craigslist <- craigslist[-which(craigslist$size >= 30000), ]
```

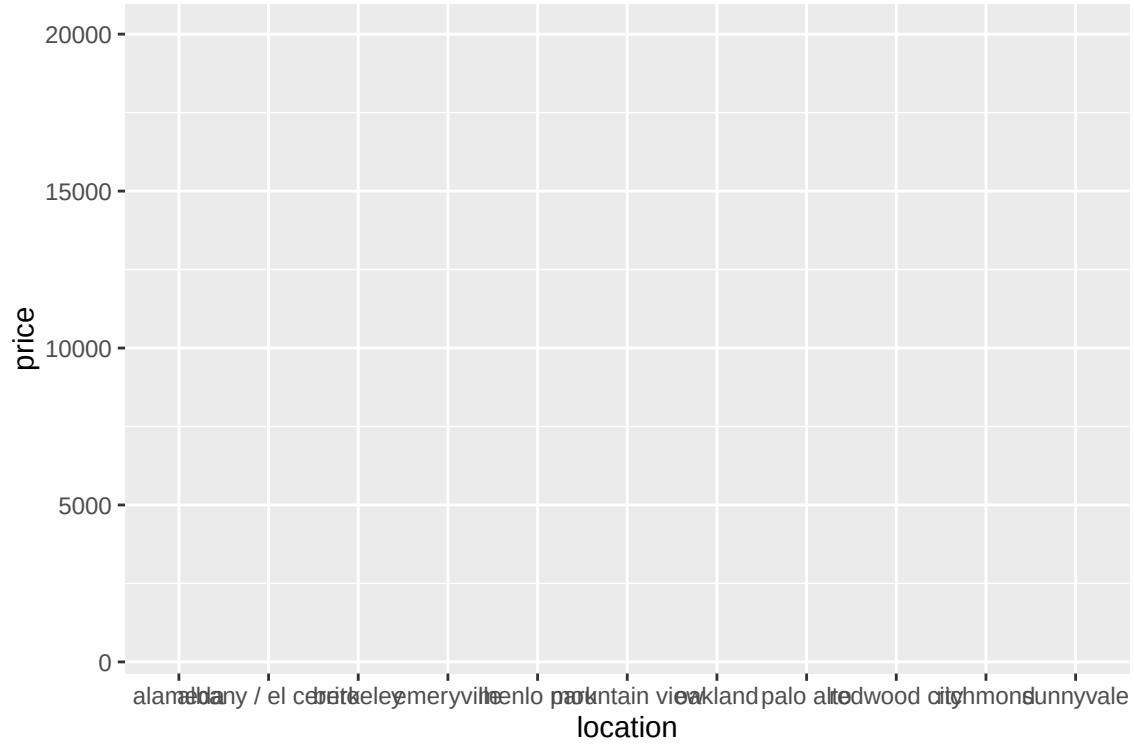
Example 1: boxplot and violin plots

First, we create a plot using `ggplot()`.

```
p <- ggplot(craigslist, # specify the data frame
            aes(x = location, # variable in the x-axis, a categorical variable in craigslist
                y = price)) # variable in the y-axis, a continuous variable in craigslist
```

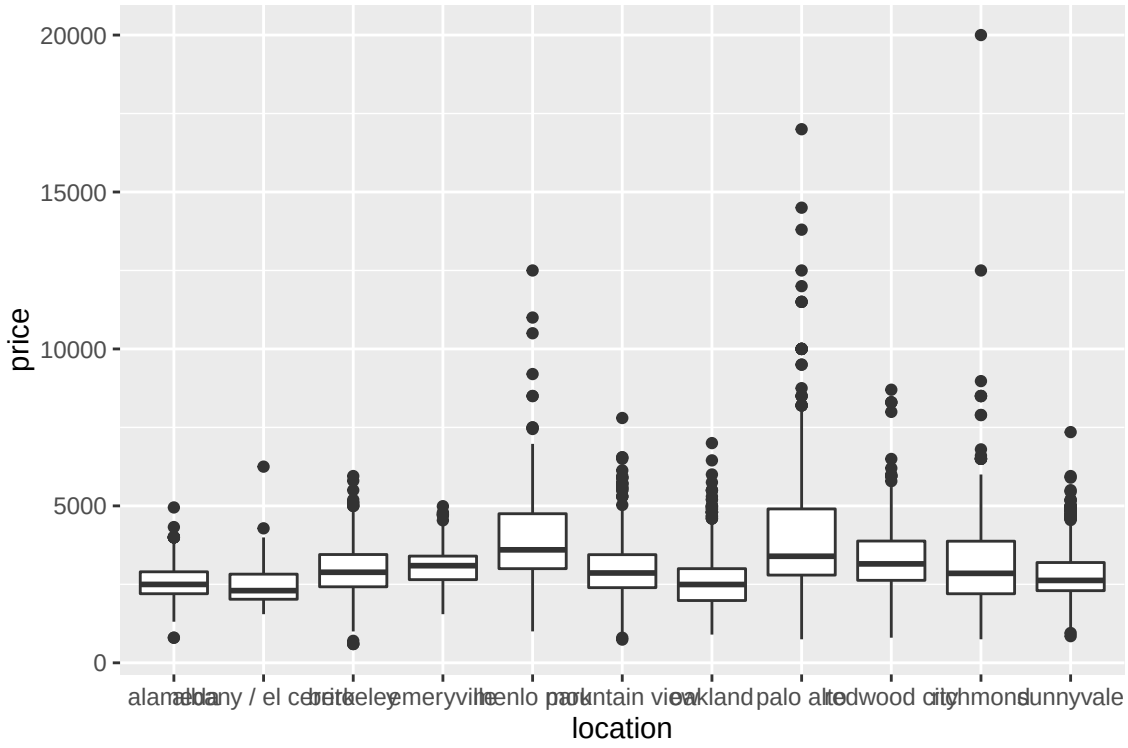
This first step initialize our plot. If you print `p`, you will see nothing other than the axis and the grid! This is because you haven't tell ggplot what to plot.

```
p
```



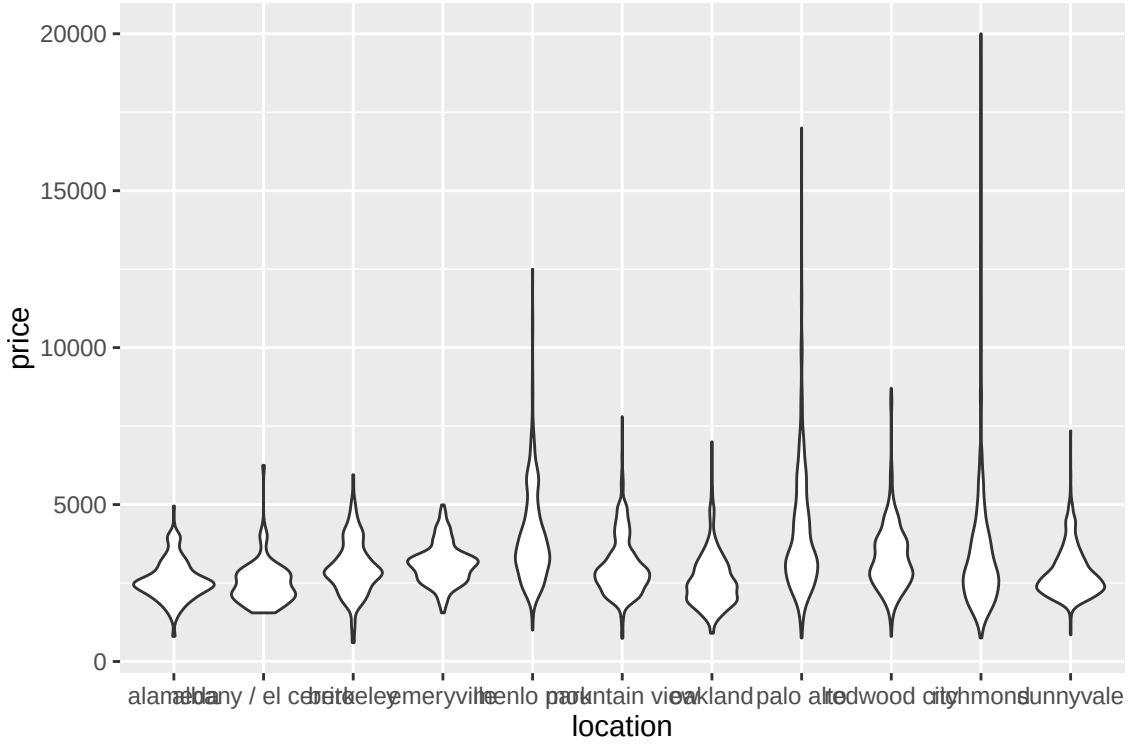
Once you initialized a plot, you are now able to add layers using various `geom` functions. For example, to add boxplots to `p`, we use `geom_boxplot`.

```
p + geom_boxplot()
```



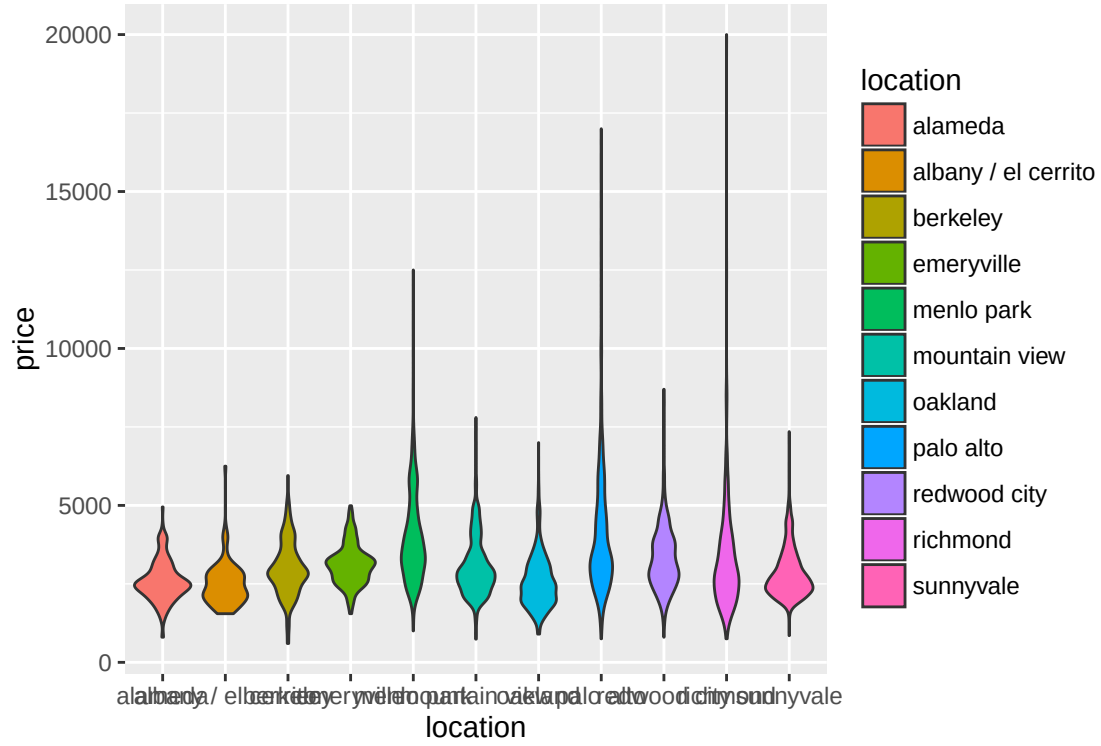
To add violin plots to `p`, we would use `geom_violin`.

```
p + geom_violin()
```



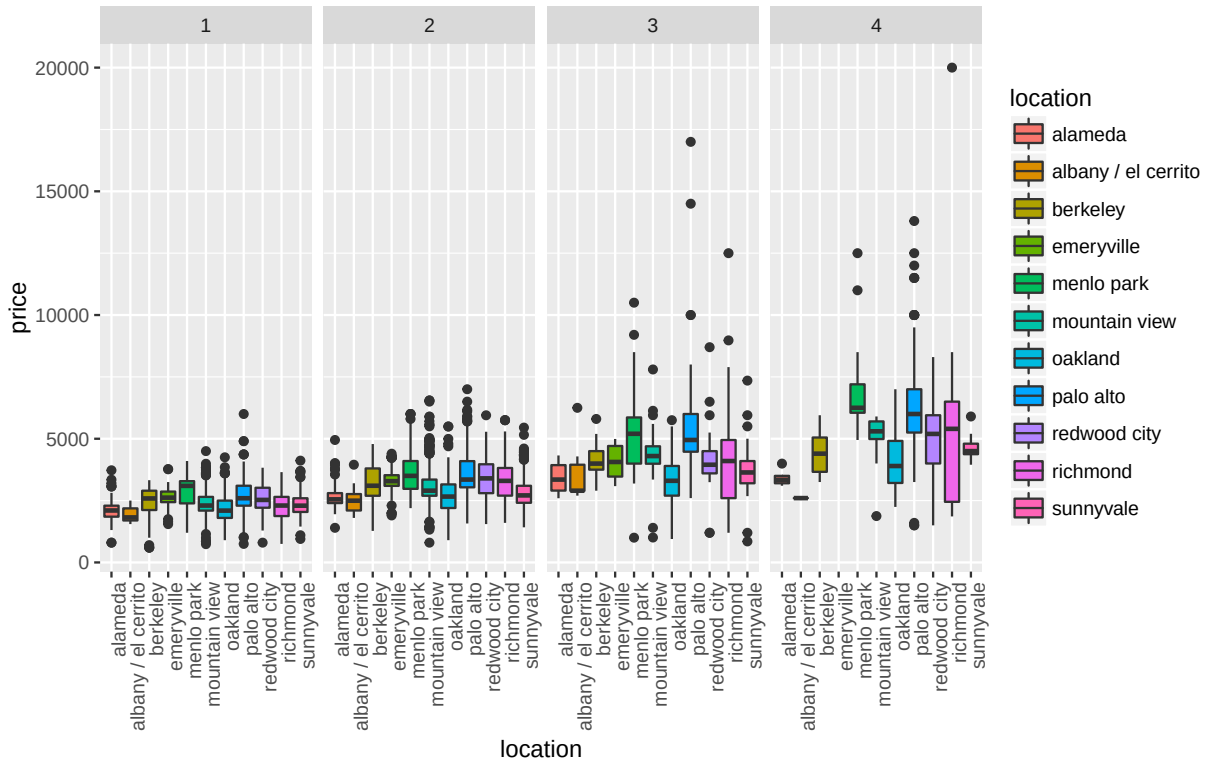
You can specify the fill color using the `fill` argument, `ggplot` will automatically create the legend for you.

```
p + geom_violin(aes(fill = location))
```



`ggplot` makes life easier for multiple categories. (This may be useful for your project 1, since you need to handle both disease treatment and urban/regions.)

```
p + geom_boxplot(aes(fill = location)) +
  facet_grid(.~brs) + # use facet by number of bedrooms
# you can do facet_grid(brs~.) as well, which gives you 4 rows of plots instead of 4 columns
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) # rotate the x labels
```



Example 2: Histogram and Density plots

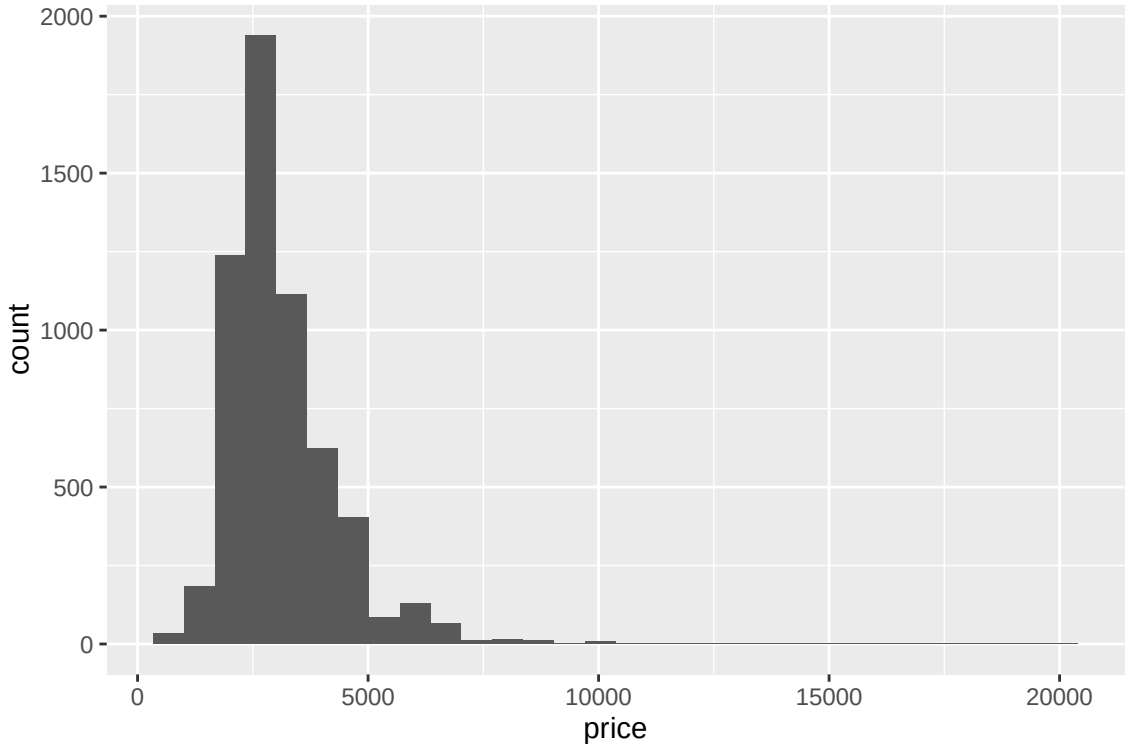
In histograms and density plots, the variable in the x-axis is from the data frame and the y-axis is the counts or the density. In such cases, we only need to specify `x` when initializing a plot with `ggplot`

```
p2 <- ggplot(craigslist,
             aes(x = price))
```

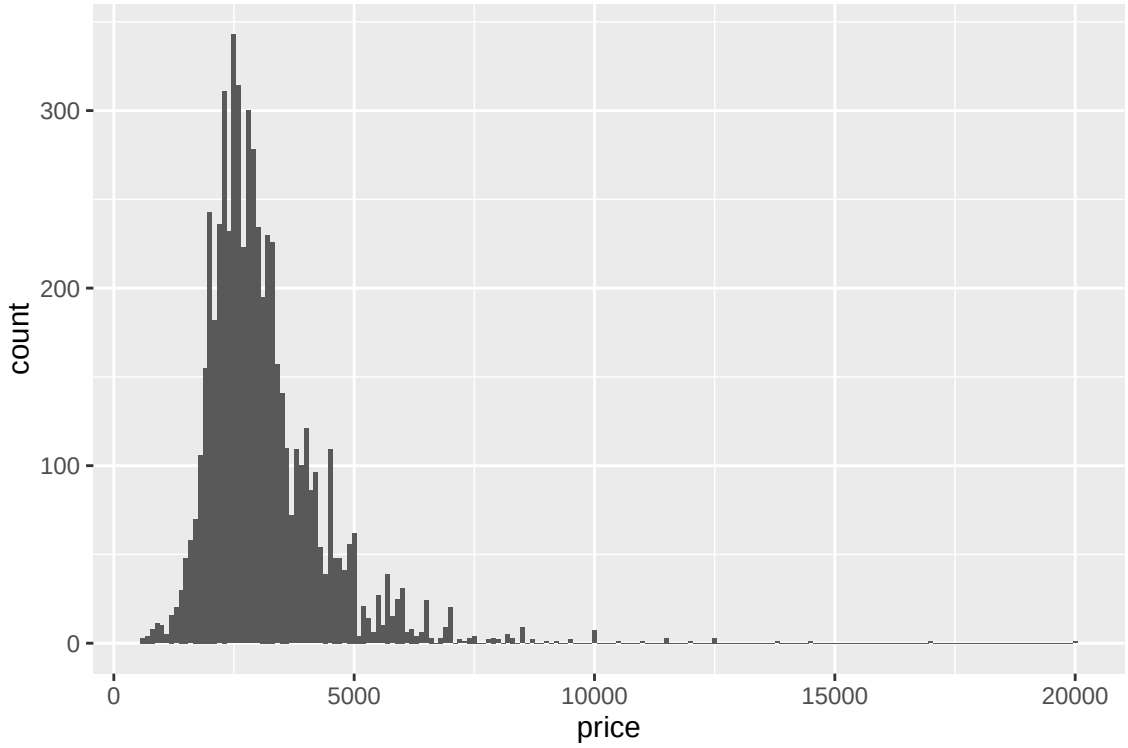
To add histogram to `p2`, we would use `geom_histogram`.

```
# A frequency histogram with the default bin width.
p2 + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

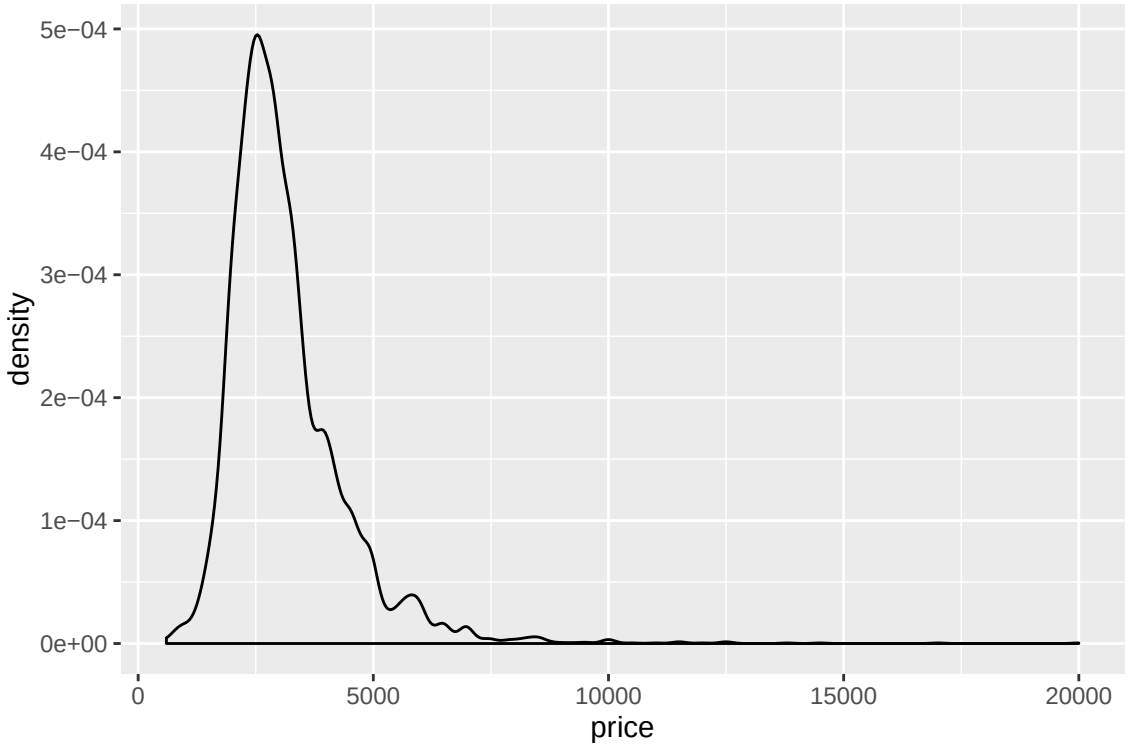



```
# A frequency histogram with bin width equal to 100
p2 + geom_histogram(binwidth = 100)
```



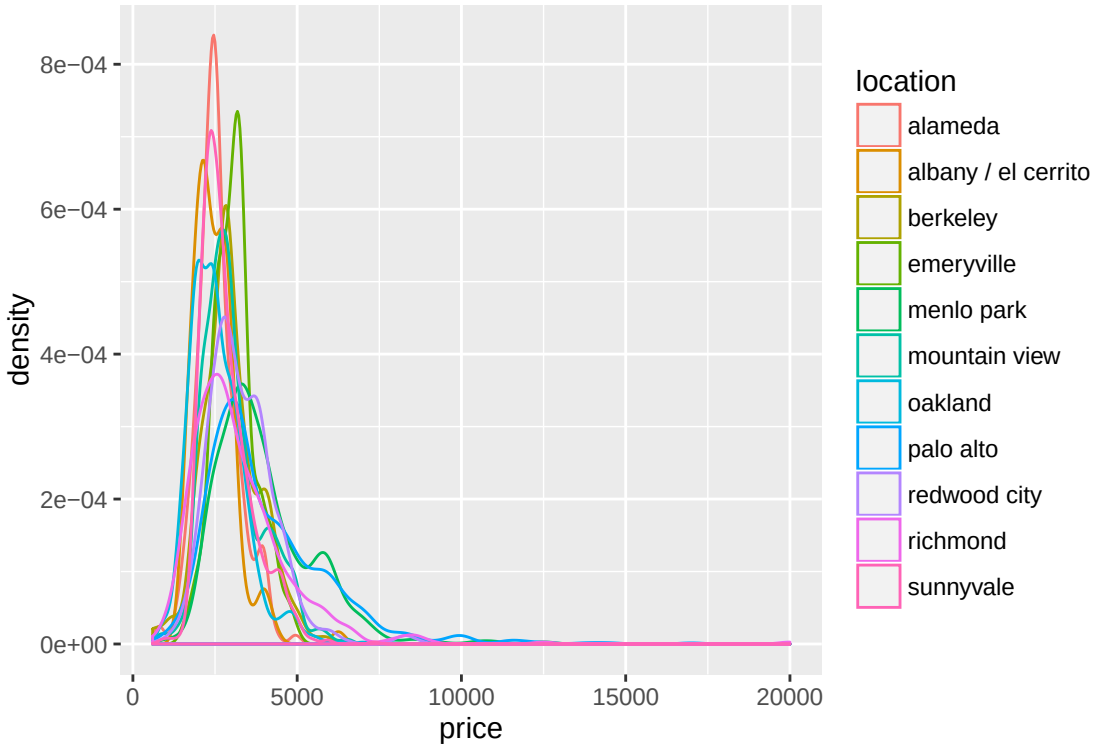
To add density to `p2`, we would use `geom_density`.

```
p2 + geom_density()
```



We could also plot densities by group.

```
p2 + geom_density(aes(colour = location))
```



Example 3: scatter plot and smooth curves

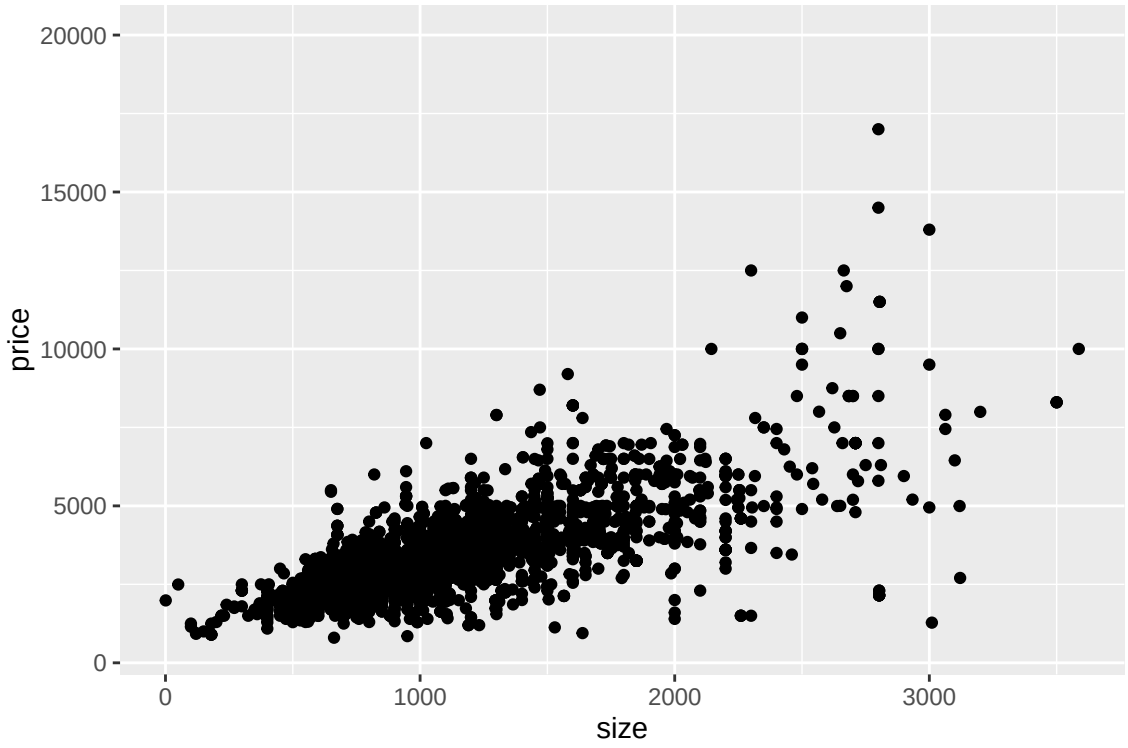
As usual, we initialize our plot by letting x-axis being the size and y-axis being the price.

```
p3 <- ggplot(craigslist, aes(x = size, y= price))
```

Add a scatter plot layer. There are missing values in the variable `size`. `ggplot` will ignore samples with missing values automatically.

```
p3 + geom_point()
```

```
## Warning: Removed 1492 rows containing missing values (geom_point).
```

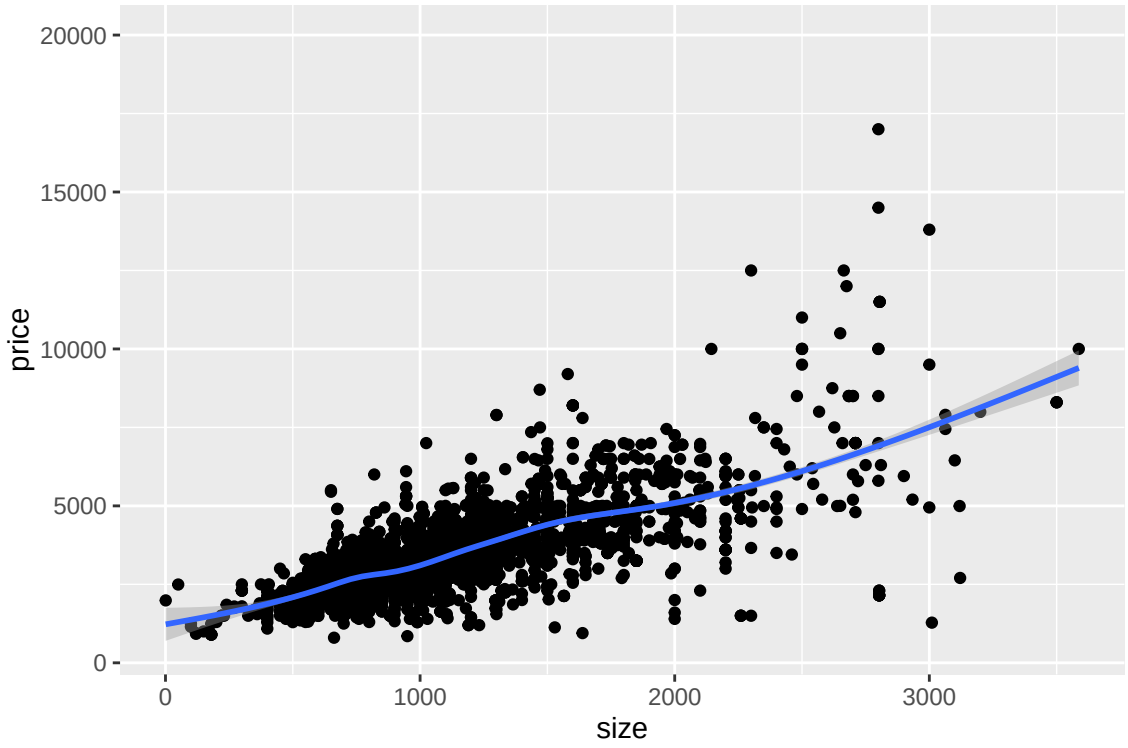


Add a smooth curve layer.

```
p3 + geom_point() + geom_smooth()
```

```
## Warning: Removed 1492 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 1492 rows containing missing values (geom_point).
```

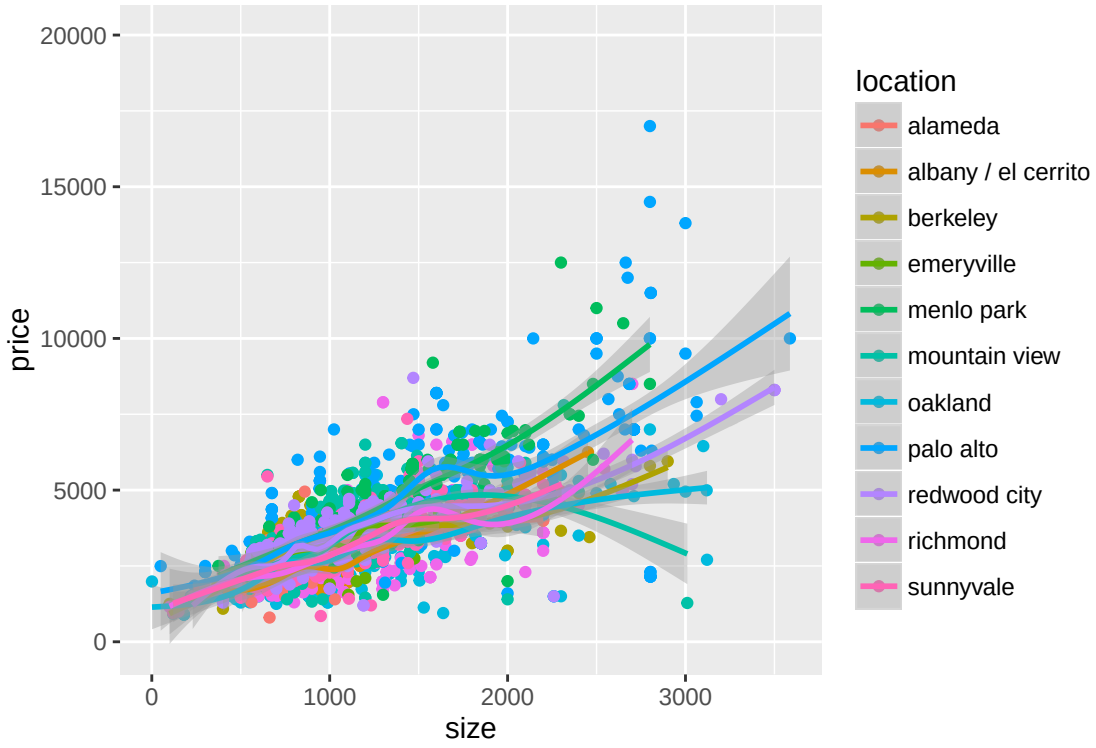


Plot the curve by group.

```
p3 + geom_point(aes(colour=location)) + geom_smooth(aes(colour=location))
```

```
## Warning: Removed 1492 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 1492 rows containing missing values (geom_point).
```



Well, this is a little messy, try the following exercise.

Exercise 3

Draw the scatter plot and smooth curve. Use `facet_grid` to group by location. Arrange these plots so that they are plotted side by side. Leave the points black and color the smooth curve by group using the `location` variable.

Example 4: Organize multiple plots in one single plot

One equivalence of `par(mfrow) = c(, )` in `ggplot` is `grid.arrange()` function in the package `gridExtra`.

```
library(gridExtra)
```

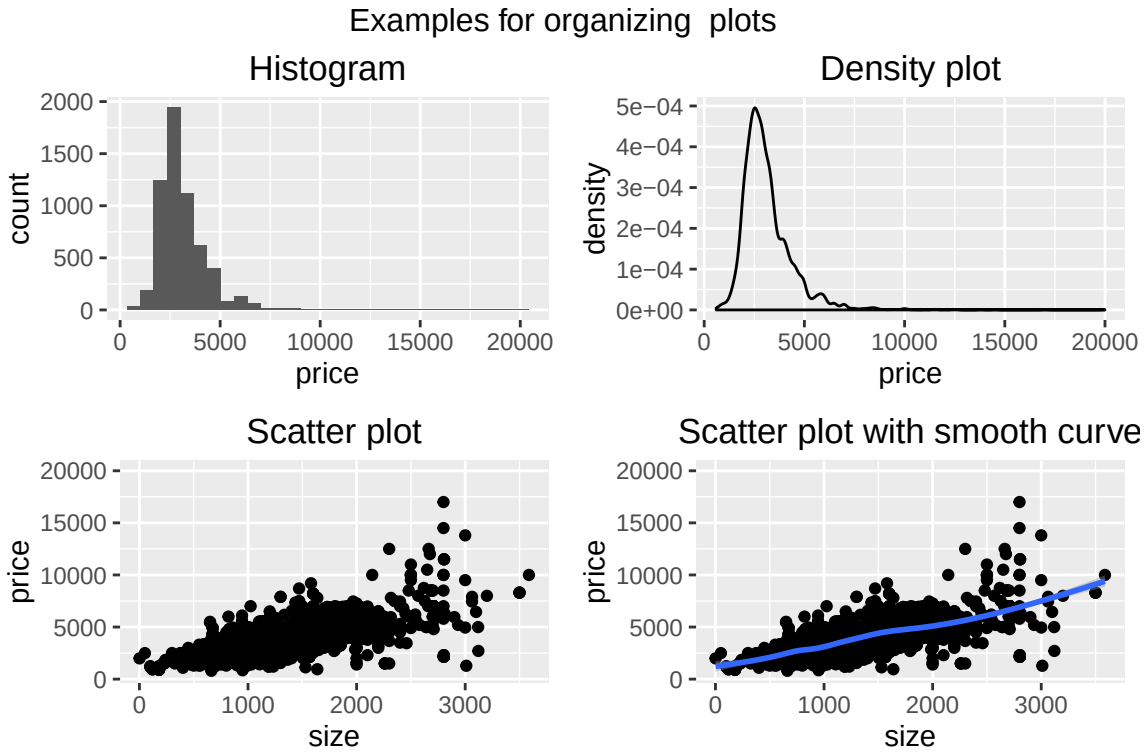
```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:Biobase':
##
##     combine
```

```
## The following object is masked from 'package:BiocGenerics':
##
##     combine
```

```
grid.arrange(p2 + geom_histogram() + ggtitle("Histogram"),
             p2 + geom_density() + ggtitle("Density plot"),
             p3 + geom_point() + ggtitle("Scatter plot"),
             p3 + geom_point() + geom_smooth() + ggtitle("Scatter plot with smooth curve"),
             nrow = 2, ncol = 2, # plot 2 by 2 grid
             top = "Examples for organizing  plots")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## Warning: Removed 1492 rows containing missing values (geom_point).
## Warning: Removed 1492 rows containing non-finite values (stat_smooth).
## Warning: Removed 1492 rows containing missing values (geom_point).
```



- Reference: Build a plot layer by layer